Reversible Data Hiding Technique using AMBTC based Bitmap Manipulation

P. Uma
Research Scholar
Department of Computer Science
Mother Teresa Woman's University, Kodaikanal

S. Vimala
Associate Professor
Department of Computer Science
Mother Teresa Woman's University, Kodaikanal

ABSTRACT

In this research work, it has been proposed to use images to hide the secret data as it is difficult to extract secret data from the image. Using images to hide and transmit secret data, leads to additional overhead of increased cost associated with storage and transmission cost, as the image requires more space for storage. However, hiding data inevitably destroys the host image, even though the distortion is imperceptible. To overcome such drawbacks, image compression techniques are used for both, hiding the secret data and to reduce the storage and transmission cost. To enhance the hiding capacity and maintain the quality of the host image after embedding hidden data, we present a high payload reversible data hiding scheme that is based on the Absolute Moment Block Truncation Coding (AMBTC) compression domain. We exploit the feature of inter block redundancy in an AMBTC compressed image to improve the coding efficiency of compressed images. Normally, the AMBTC technique transforms an input image into a set of blocks and each block in turn is transformed into a set of a bit-plane and two quantizers. But in the proposed method, these blocks are categorized into Shade and Edge blocks based on the magnitude of amplitude values. For a Shade block, two values are stored, one being the block mean and the other value being the secret data. Two bytes of secret data is stored creating an illusion that the bitplane is stored. For Edge blocks, high mean and low mean are stored as two quantizers and a respective Bitplane is generated. The shade block helps in embedding 3 bytes of secret data leading to higher embedding capacity and increases the complexity of identifying the existence of secret data. It also compresses the stego-image without much degradation in the visibility of stego image.

Keywords

AMBTC Compression, Embedding Capacity, Reversible Data Hiding, Stego image.

1. INTRODUCTION

Reversible data hiding (RDH) is a technique which embeds secret data into a cover image and can extract the embedded data and recover the original image with lossless. Cover image refers to the image used for carrying the embedded data. Embedded data is known as payload and the image with embedded data is called stego image [1]. Reversible data hiding technique is used in some special applications such as Military images, Medical Images and Forensics, where the exact recovery of the original cover image is inevitable. Data hiding techniques can be carried out in three domains [2] namely; Spatial Domain [3, 4], Frequency Domain [5, 6, 7], and Compression Domain [8, 9, 10]. In the Spatial Domain based data hiding techniques, the secret data is embedded by modifying the pixel values directly. These techniques are the simplest and have the least computational complexity. In most of the spatial data hiding techniques, least significant bits (LSB) [11] and pixel value differences (PVD) [12] are exploited for embedding the secret data. In frequency domain based data hiding techniques, the original image is first transformed into frequency coefficients and then the secret data is embedded as part of transformed frequency coefficients. In compression domain based data hiding techniques, the original image is first compressed into compressed codes using any available image compression methods such as Joint Photographic Experts Group (JPEG) [13], Vector Quantization (VQ) [8], Graphics Interchange Format (GIF) [14], and Block Truncation Coding (BTC) [15] and then the secret data is embedded as part of the compressed codes. The secret data that is hidden as part of the compressed image is not easily visible and hackers or attackers will not usually be wary on stego-images.

Over the last few years, many hiding schemes have been proposed based on Block Truncation Coding (BTC) [16–22], which has been the most efficient, simplest and fastest compression method. In 2008, Chang et al. proposed a novel reversible data hiding method using BTC that compresses color images. It requires three bitmaps and three pairs of quantization levels for reconstruction. Genetic algorithm (GA) is applied to find an approximate optimal common bitmap to replace the original three bitmaps to improve the coding efficiency. The secret data is embedded in the common bitmap and the quantization levels of each block use the properties of side matching and the order of quantization levels to achieve reversibility [9]. In 2010, Chen et al. proposed an improved data hiding scheme to embed the secret data in compressed bit streams and the quality of the image is also maintained even after embedding [16]. The difference of quantization levels for each block is determined whether the only 1 bit of secret data is to be hidden or to toggle bits in the bitmap to hide more bits. This scheme is very simple and does not require complex computations. In 2011, Li et al. presented the histogram shifting and bitplane flipping technique based on BTC compressed images to improve the hiding capacity and maintaining image quality [17]. In 2013, Sun et al. presented a novel BTC-based reversible hiding scheme by adopting a joint neighbor coding technique to embed the secret data into quantization levels [18]. In 2015, Lin et al. presented a high payload reversible data hiding scheme based on Absolute Moment Block Truncation Coding (AMBTC). It discovers the redundancy in a block of AMBTC compressed images to determine if a block is embeddable or non-embeddable. In this scheme, four disjoint sets are created for embeddable blocks to embed secret data using different combinations of the mean and standard deviation [21]. This scheme adopts both spatial and compression domain because this method utilizes the concept of AMBTC and they do not compress the image so that the stego image is not the BTC codestream. In 2019, Lin et al. proposed BTC based reversible data hiding scheme without Blocking Effect problem. In this scheme, Canny edge detector is used to classify a block in a cover image into Edge block and Non-Edge block. Then Zero-Point Fixed Histogram Shifting (ZPF-HS) was applied to embed the secret data into compressed code [22].

The following sections are organized such that Section-2 introduces the AMBTC compression technique and Ou and Sun's Scheme which forms the basis of our proposed reversible hiding scheme; Section-3 introduces the proposed method; Experimental results are discussed in Section-4 and finally, conclusions are presented in Section-5.

2. RELATED WORKS

2.1 Absolute Moment Block Truncation Coding [23]

In 1984, Lema and Mitchell proposed the method of Absolute Moment Block Truncation Coding (AMBTC) to compress images. AMBTC is a lossy image compression method which requires low computation cost. AMBTC is suitable for real-time embedded system applications and it can obtain acceptable image quality. In AMBTC, the input image is split up into multiple blocks of size 4 x 4 pixels. For each block, the Mean $(\bar{x})$ is computed using (1). Two quantization levels HMean and LMean are computed using (2) and (3).

$$Mean = \frac{1}{16}\sum_{i=1}^{16} x_i \qquad (1)$$

$$LMean = \frac{1}{p}\sum_{x_i<\bar{x}} x_i \qquad (2)$$

$$HMean = \frac{1}{q}\sum_{x_i\geq\bar{x}} x_i \qquad (3)$$

where x_i is individual pixel value, p is the number of pixels whose values are less than $\bar{x}$ and q is number of pixels whose values are greater than $\bar{x}$. A Bitmap for each block is generated using (4).

$$BM = \begin{cases} 1 & if\ x_i \geq \bar{x} \\ 0 & otherwise \end{cases} \qquad (4)$$

As a result of Encoding, the compressed image block is represented as a set of three components viz. $\{LMean,\ HMean,\ BM\}$. In decoding procedure, the image block can be reconstructed by substituting LMean and HMean for 1 and 0 respectively in bitmap BM.

2.2 Ou and Sun's Scheme [24]

Ou and Sun's scheme was proposed as an improved steganography scheme based on AMBTC. In this scheme, a threshold is predefined to divide the image blocks into Shade or Edge blocks. For a shade block, the bitmap is used to embed the secret data. After embedding the secret data into bitplane, the two quantization levels are recalculated based on the new bitmap. The new quantization levels minimize the distortion of image block. For the edge blocks, secret data are embedded by exchanging the order of two quantization levels and bitmap. In edge blocks, embedding capacity is also increased without any distortion.

2.2.1 Embedding Phase

Input: Input image I of size M x N, Secret data sequence S and Threshold value (Th)

Output: A stego AMBTC compressed code (I_s).

Step1: Divide the input image block I into non-overlapping 4 x 4 blocks.

Step 2: For each block, calculate two quantization values and bitmap are generated using AMBTC method.

Step 3: Compute the absolute difference (d_i) between the two quantization values (a_i, b_i) such that, $d_i = |a_i - b_i|$

Step 4: if $d_i > Th,$ the block is edge block, fetch the one bit s1 from S. If s1 is equal to 1, the bitmap Bi is inversed and two quantization levels are exchanged. Such that original compressed code $(a_i,\ b_i,\ B_i)$ are transformed into $(b_i,\ a_i,\ \overline{B_i})$ and then add $(b_i,\ a_i,\ \overline{B_i})$ into I_s. If s1 is equal to 0, no operation is performed and then adds original compressed code $(a_i,\ b_i,\ B_i)$ into I_s.

Step 5: if $d_i \leq Th,$ the block is shade block. Fetch the 16 bits s2 from S. the bitmap is replaced with s2 and form the new bitmap $B_i^{'}$ and recalculate the two quantization levels $(a_i^{'},\ b_i^{'})$ based on new bitmap $B_i^{'}$.

Step 6: $If\ |a_i^{'} - b_i^{'}| \leq Th,$ recalculate quantization levels are maintained the smoothness of the block and then add $(a_i^{'},\ b_i^{'},\ B_i^{'})$ into I_s. Otherwise $If\ |a_i^{'} - b_i^{'}| > Th,$ two old quantization levels (a_i, b_i) maintain the smoothness of the block and then add $(a_i,\ b_i,\ B_i^{'})$ into I_s.

Step 7: Repeat steps 2 to 6 until all the image blocks are processed.

2.2.2 Extracting Phase

Input: A Stego compressed codes I_s size of $\frac{M}{4} x \frac{N}{4}$ trios $(a_i,\ b_i,\ B_i)$ where $i = 1,2,\ldots,\frac{M}{4} x \frac{N}{4}$ and threshold Th.

Output : Secret bit sequences (SE)

Step 1: For each trio $(a_i,\ b_i,\ B_i)$ in I_s, compute absolute difference d_i, such that $d_i = |a_i - b_i|$.

Step 2: $if\ d_i > Th,$ one bit s1 is extracted. $if\ a_i > b_i,$ s1 can be 1, otherwise s1 can be 0 and then add extracted bit s1 into SE.

Step 3: $if\ d_i \le Th,$ 16 bits s2 is extracted from the bitmap B_i . Then add s2 into SE.

Step 4: Repeat steps 1 to 3 until all the blocks are processed.

3. PROPOSED METHOD

The proposed method consists of three stages; i. AMBTC based compression, ii. Data Hiding and iii. Data Recovery.

3.1 The AMBTC Compression Stage

The input image is compressed using AMBTC technique. Given an M x N sized gray scale original image X, the following are the steps followed to transform the image into a compressed form based on AMBTC.

Step 1: The original gray scale image of size M x N is divided into non-overlapping blocks of size 4 x 4.

Step 2: Calculate the mean of the block using the equation (1).

Step 3: Compute the HMean and LMean values using (2) and (3).

Step5: Set the threshold (TH).

Step6: Find the difference D between HMean and LMean.

Step7: If $D <= TH,$ the block is Shade block and store only the Mean value.

Step8: if $D > TH,$ the block is Edge block. Generate the Bitmap (BM) using (4) and compute the HMean and LMean values.

Step9: Repeat the steps 3 to 8 for each block.

3.2 The Data Hiding Stage

The secret data is hidden as part of the compressed image. The proposed method is a reversible data hiding algorithm. Before embedding, the secret data bit sequence $S = \{s_1, s_2, ...\}$ undergoes permutation operation in order to enhance the security by increasing the identification complexity. In AMBTC, all input blocks are treated equally and for each block, a set of two quantizers and a Bitmap (BM) of size 16 bits are generated. In the proposed method, an indicator bit (1/0) is used to represent a shade block (0) and an edge block (1).

Each compressed block is transformed into a bit sequence of size 49 bits as follows:

$$[IndicatorBit \,\|\quad Q1 \,\|\quad Q2 \,\|\quad BP]$$

For a shade block, two values are stored. The first value is the Mean of the block (Q1) and the second value is the 8-bit secret data (Q2). As a Bitmap is not necessary for a shade block, a 32 bit secret data is stored as Bitmap (BM) which increases the embedding capacity. Hence in a shade block, a secret data of 40 bits is stored out of 49 bits.

For an edge block, HMean (Q1) and LMean (Q2) and a 32-bit Bitmap are generated. HMean and LMean are block-specific. In the Bitmap, out of 2 bits, the first bit indicates the nature of the input image's pixel value and the second bit represents the secret data. In edge blocks, 16 bits of secret data is stored out of 49 bits.

The formation of BM for an edge block is as follows:

$$\begin{bmatrix} b_1 & b_2 & b_3 & b_4 \\ b_5 & b_6 & b_7 & b_8 \\ b_9 & b_{10} & b_{11} & b_{12} \\ b_{13} & b_{14} & b_{15} & b_{16} \end{bmatrix} + \begin{bmatrix} s_1 & s_2 & s_3 & s_4 \\ s_5 & s_6 & s_7 & s_8 \\ s_9 & s_{10} & s_{11} & s_{12} \\ s_{13} & s_{14} & s_{15} & s_{16} \end{bmatrix} =$$

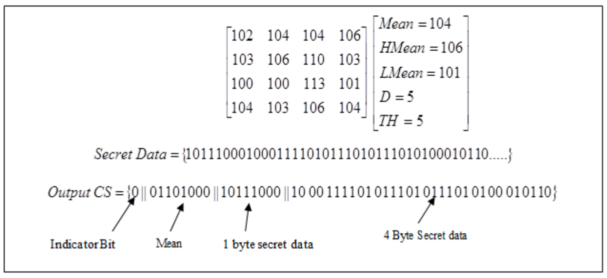
$$\begin{bmatrix} b_1s_1 & b_2s_2 & b_3s_3 & b_4s_4 \\ b_5s_5 & b_6s_6 & b_7s_7 & b_8s_8 \\ b_9s_9 & b_{10}s_{10} & b_{11}s_{11} & b_{12}s_{12} \\ b_{13}s_{13} & b_{14}s_{14} & b_{15}s_{15} & b_{16}s_{16} \end{bmatrix}$$

3.3 Second Level of Compression

To reduce the Bit rate, LMean value is subtracted from HMean and the difference is stored as LMean, it requires only less number of bits. This leads to further reduction in image data. By subtracting the difference from HMean, LMean can be recovered later while reconstructing the image from compressed data.

For example, HMean = 236, LMean = 225, D = 236 – 225 = 11. Generally, we need 8 bits to store HMean and LMean values. But to store the difference in this case, we need only 4 bits.

Embedding of secret data as part of Shade block is explained in Fig. 1.



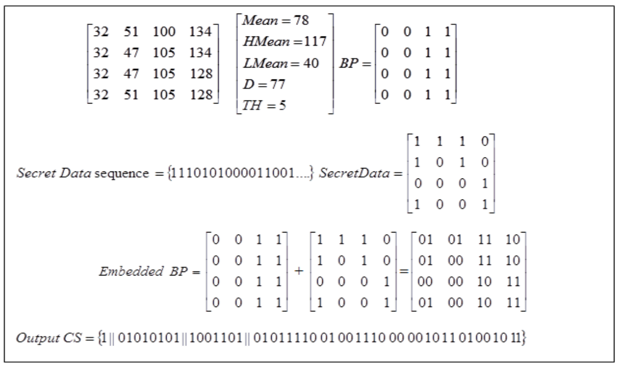
$$\begin{bmatrix} 102 & 104 & 104 & 106 \\ 103 & 106 & 110 & 103 \\ 100 & 100 & 113 & 101 \\ 104 & 103 & 106 & 104 \end{bmatrix} \begin{bmatrix} Mean = 104 \\ HMean = 106 \\ LMean = 101 \\ D = 5 \\ TH = 5 \end{bmatrix}$$

$$Secret\ Data = \{10111000100011110101110101110101000010110.....\}$$

$$Output\ CS = \{0 \,\|\, 01101000 \,\|\, 10111000 \,\|\, 10\ 00\ 111101\ 011101\ 011101\ 0100\ 010110\}$$

Fig. 1. Example of Secret data embedding in Shade Block

Embedding of secret data as part of Shade block is explained in Figure 1 and Embedding of secret data as part of Edge Block is explained in Figure 2.



$$\begin{bmatrix} 32 & 51 & 100 & 134 \\ 32 & 47 & 105 & 134 \\ 32 & 47 & 105 & 128 \\ 32 & 51 & 105 & 128 \end{bmatrix} \begin{bmatrix} Mean = 78 \\ HMean = 117 \\ LMean = 40 \\ D = 77 \\ TH = 5 \end{bmatrix} BP = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$Secret\ Data\ \text{sequence} = \{1110101000011001....\}\ SecretData = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$Embedded\ BP = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 01 & 01 & 11 & 10 \\ 01 & 00 & 11 & 10 \\ 00 & 00 & 10 & 11 \\ 01 & 00 & 10 & 11 \end{bmatrix}$$

$$Output\ CS = \{1 \,\|\, 01010101 \,\|\, 1001101 \,\|\, 01011110\ 01\ 001110\ 00\ 001011\ 010010\ 11\}$$

Fig. 2. Example of Secret data embedding in Edge Block

3.4 Data Extraction and Recovery Stage

In case of a Shade block, the second quantizer (Q2) and the Bitplane (BP) give the secret data. The compressed image block is reconstructed by filling all pixel values with the Mean value (Q1). In Edge block, 16 bits of secret data is retrieved by extracting the second bit from each element of Bitplane.

The current block is replaced with HMean and LMean based on the original Bitplane using (5).

$$RB(i,j)=\begin{cases} LMean & if\ BP(i,j)==0 \\ HMean & if\ BP(i,j)==1 \end{cases} \tag{5}$$

The flow of Data embedding stage is explained in Fig. 3.

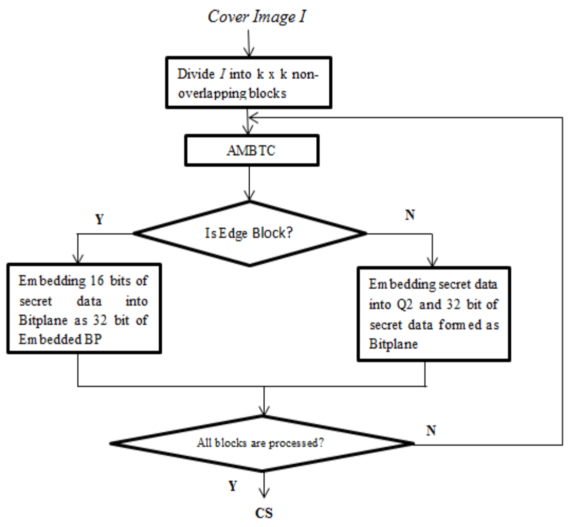

Cover Image I
Divide I into k x k non-overlapping blocks
AMBTC
Is Edge Block?
Y
N
Embedding 16 bits of secret data into Bitplane as 32 bit of Embedded BP
Embedding secret data into Q2 and 32 bit of secret data formed as Bitplane
All blocks are processed?
N
Y
CS

Fig. 3. The flowchart of Data Embedding Stage

For example,

In shade block,

$$OutputCS=\begin{Bmatrix} 0\,||\,01101000\,||\,10111000\,|| \\ 10001111010111010111010100010110 \end{Bmatrix}$$

In the above output code stream, indicator bit is 0, it is a shade block. We retrieve Q1 as Mean and the value is 104. Q2 and 32-bit plane are secret data.

$$SecretData=\begin{Bmatrix} 10111000100011110101110101111 \\ 010100010110 \end{Bmatrix}$$

$$RB=\begin{bmatrix} 104 & 104 & 104 & 104 \\ 104 & 104 & 104 & 104 \\ 104 & 104 & 104 & 104 \\ 104 & 104 & 104 & 104 \end{bmatrix}$$

In Edge block,

$$OutputCS=\begin{Bmatrix} 1\,||\,01110101\,||\,1001101\,|| \\ 01011110010011100000101101001011 \end{Bmatrix}$$

Indicator bit is 1, it is an edge block. We retrieve Q1 and Q2 as HMean and D respectively.

HMean=117, D=77, LMean=117-77=40.

$$Embedded\ BP=\begin{bmatrix} 01 & 01 & 11 & 10 \\ 01 & 00 & 11 & 10 \\ 00 & 00 & 10 & 11 \\ 01 & 00 & 10 & 11 \end{bmatrix}$$

$$Secret\ Data=\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$BP=\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$RB=\begin{bmatrix} 40 & 40 & 117 & 117 \\ 40 & 40 & 117 & 117 \\ 40 & 40 & 117 & 117 \\ 40 & 40 & 117 & 117 \end{bmatrix}$$

The workflow of data extraction and recovery stage is explained in Figure 4.

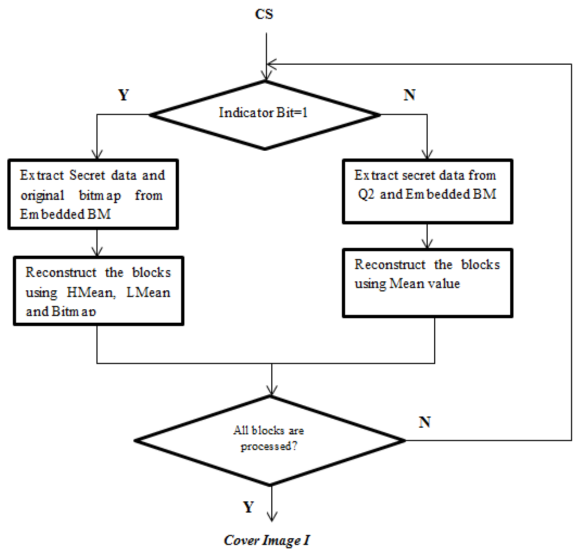

CS
Y
N
Indicator Bit=1
Extract Secret data and original bitmap from Embedded BM
Extract secret data from Q2 and Embedded BM
Reconstruct the blocks using HMean, LMean and Bitmap
Reconstruct the blocks using Mean value
All blocks are processed?
N
Y
Cover Image I

Fig 4. The workflow of Data Extraction and Recovery Stage

4. RESULTS AND DISCUSSIONS

All experiments are performed with the benchmark gray scale images of size 512 x 512 pixels. The input images are Airplane, Lena, Boats, Bridge, Man, Peppers, Sailboat, Zelda, Girl and Couple. Fig. 5 shows the input images taken for the study. The algorithms are implemented using MATLAB R2014b. The Hardware used is, the Intel® Pentium® 1.90 GHZ processor with 8GB RAM.

Fig. 5. Input images taken for the study

In our proposed method, the secret data is embedded into compressed image and the image is called Stego Image. Then we extract the embedded secret data from Stego Image thereby leading to original Cover Image. The Peak Signal to Noise Ratio (PSNR) is used to measure the quality of the compressed image using (6).

$$PSNR = 10\log_{10}\left[\frac{255^2}{MSE}\right] dB \tag{6}$$

where, MSE is the Mean Square Error which is used to measure the error between the original image and compressed image and is computed using (7).

$$MSE = \frac{1}{MxN}\sum_{i=1}^{M}\sum_{j=1}^{N}(I(i,j) - I'(i,j))^2 \tag{7}$$

where, M is the number of rows and N is number of columns. I(i, j) and I'(i, j) denote the Original image and Compressed image respectively.

The bitrate is used to analyze the coding efficiency of any Compression algorithm. The bitrate is computed using the equation (8), where CS represents the length of compression code, M x N is the size of the original image. The bitrate measured is represented in terms of bits per pixel (bpp).

$$bitrate = \frac{CS}{MxN}(bpp) \tag{8}$$

To illustrate the performance of our proposed method, the results of our scheme with various threshold values ranging between 5 to 25 are shown in Table I. Input images of size 512 x 512 are taken for the study and the optimal block size is taken as 4 x 4 pixels. The proposed method is tested with five different images and the average embedding capacities obtained are 366589, 452393, 507274, 548740, 571092 and the average PSNR and BPP of the images are 39.64, 38.24, 37.48,36.90, 36.48 and 2.87, 2.94, 2.99, 3.02, 3.03 respectively.

The performance of the proposed scheme is compared with the existing schemes (reversible schemes) proposed by Chang et al.[9], Li et al.[17], Sun et al.[18], Lin et al.[21] and Chan et al.[22] in terms of embedding efficiency (EE) and is shown in Table II.The Embedding Efficiency (EE) is computed using the equation (15).

$$EE = \frac{Embedding\ Capacity}{\|CS\|} x100 \tag{15}$$

where ||CS|| is the size of the compressed image in terms of bits.

Table I. Comparison of PSNR, Embedding Capacity and BPP of Proposed method with Various Thresholds

Images	Parameters	THRESHOLD Range (5-25)				
		TH=5	TH=10	TH=15	TH=20	TH=25
Lena	PSNR	37.58	36.96	36.36	35.86	35.48
	Capacity	417688	518104	560704	584392	599776
	BPP	2.90	2.98	3.02	3.03	3.04
F16	PSNR	37.33	37.01	36.62	36.23	35.93
	Capacity	462544	520696	549880	569536	581752

	BPP	2.96	3.00	3.02	3.03	3.04
Sail Boat	**PSNR**	35.13	34.68	34.24	33.91	33.62
	Capacity	334384	445288	499072	528568	551272
	BPP	2.85	2.95	2.99	3.01	3.02
Girl	**PSNR**	37.91	37.32	36.30	35.45	34.89
	Capacity	379552	458320	528640	571048	595336
	BPP	2.90	2.96	3.00	3.03	3.04
Toys	**PSNR**	37.98	37.63	37.38	37.05	36.63
	Capacity	487120	541384	556336	568384	581488
	BPP	2.95	3.00	3.01	3.02	3.03
Barbara	**PSNR**	42.29	41.34	36.91	33.83	33.75
	Capacity	262216	309064	473368	649720	653416
	BPP	2.67	2.73	2.91	3.06	3.06
Baby	**PSNR**	32.83	32.77	32.49	32.01	31.53
	Capacity	273928	298072	346720	406480	461656
	BPP	2.87	2.89	2.92	2.96	2.99
Baboon	**PSNR**	32.75	32.62	32.32	32.00	31.71
	Capacity	275224	325216	380560	422368	456760
	BPP	2.85	2.89	2.94	2.96	2.98
Boats	**PSNR**	35.98	35.39	34.77	34.24	33.88
	Capacity	320440	448192	511984	549520	572032
	BPP	2.86	2.96	3.00	3.02	3.03
Bridge	**PSNR**	33.47	33.35	33.02	32.55	32.09
	Capacity	278032	322660	374080	426688	472600
	BPP	2.85	2.89	2.93	2.96	2.99
Goldhill	**PSNR**	37.28	36.52	35.62	34.91	34.44
	Capacity	339856	457120	528496	569392	594160
	BPP	2.84	2.94	3.00	3.02	3.04
Girlface	**PSNR**	38.39	37.89	37.35	36.89	36.50
	Capacity	482200	548536	577768	596296	609616
	BPP	2.93	2.99	3.02	3.03	3.04
Pepper	**PSNR**	37.90	36.89	36.27	35.87	35.60
	Capacity	367936	525736	572968	593680	605968
	BPP	2.86	2.99	3.03	3.04	3.04
Tiffany	**PSNR**	39.32	38.49	37.67	37.06	36.61
	Capacity	438040	534016	577456	600520	614824
	BPP	2.93	3.00	3.03	3.04	3.05
Man	**PSNR**	35.49	35.13	34.63	34.05	33.60
	Capacity	344032	428752	481288	524896	555448
	BPP	2.85	2.93	2.97	3.00	3.02
Zelda	**PSNR**	39.64	38.24	37.48	36.90	36.48
	Capacity	402232	557128	597064	618352	631360

	BPP	2.86	3.00	3.03	3.04	3.05
Average PSNR		**36.95**	**36.39**	**35.59**	**34.93**	**34.55**
Average Capacity		**366589**	**452392.8**	**507274**	**548740**	**571092**
Average BPP		**2.87**	**2.94**	**2.99**	**3.02**	**3.03**

Table II. Comparison EE (%) for the proposed scheme and Existing Schemes

Schemes	Parameters	Chang et al. [9]	Li et al.[17]	Sun et al.[18]	Lin et al.[21]	Lin et al.[22]	Proposed
Lena	Capacity	31011	16789	64008	76671	262112	417688
	CS	524288	524288	524288	480096	2097152	759860
	EE	6%	3%	12%	16%	13%	55%
F16	Capacity	30518	17659	64008	80009	261984	462544
	CS	524288	524288	524288	474264	2097152	775529
	EE	6%	3%	12%	17%	12%	60%
SailBoat	Capacity	28766	17082	64008	84791	262096	334384
	CS	524288	524288	524288	495432	2097152	747747
	EE	5%	3%	12%	17%	13%	45%
Girls	Capacity	30962	16990	64008	108217	262128	379552
	CS	524288	524288	524288	544104	2097152	758928
	EE	6%	3%	12%	20%	13%	50%
Toys	Capacity	27870	17761	64008	70889	262112	487120
	CS	524288	524288	524288	455112	2097152	774345
	EE	5%	3%	12%	16%	13%	63%
Barbara	Capacity	30151	16755	64008	87264	262128	262216
	CS	524288	524288	524288	510048	2097152	700832
	EE	5%	3%	12%	17%	13%	37%
Average	**Capacity**	**29880**	**17173**	**64008**	**84640**	**262093**	**390584**
	CS	**524288**	**524288**	**524288**	**493176**	**2097152**	**752874**
	EE	**6%**	**3%**	**12%**	**17%**	**13%**	**52%**

It is observed from Table II, the results in terms of Embedding Capacity and Embedding Efficiency are better with proposed method when compared to that of the existing similar methods and are highlighted. An average embedding capacity of 390584 bits and embedding efficiency of 52% are obtained with the proposed method.

5. CONCLUSION

A novel reversible data hiding method using Absolute Moment Block Truncation Coding based on Edge and Shade block approach is presented in this paper. Hence adaptive embedding is possible based on the type (Shade or Edge) of the input image block. Our proposed method is suitable for hiding large volume of information in multimedia. The maximum embedding capacity obtained with the existing method [22] is 1,28,491 bits less than that of proposed, which is a significant improvement. Similarly, the maximum average Embedding Efficiency obtained with the existing method [21] is 35% less than that of the proposed method. The proposed method, hence outperforms than all the existing similar methods mentioned in this both in terms of Embedding Capacity and Embedding Efficiency. Instead of using 16 bits out of 32 bits of Bitplane of an Edge block for embedding the secret data, the embedding capacity can further be improved by trying to make use of all 32 bits in the future without loss of quality in the Stego Image.

6. REFERENCES

[1] M.S.Subhedar and V.H.Mankar, “Current status and key issues in image steganography: A survey”, Computer Science Review (2014).

[2] Rajeev Kumar and Ki-Hyun Jung, “A Systematic survey on block truncation coding based data hiding techniques”, Multimedia Tools and Applications, July 2019.

[3] X.Liao, Q.Wen, and J.Zhang, “Improving the adaptive steganographic methods based on modulus function”,

IEICE Transactions on Fundamentals of Electronics, Communication and Computer Science, Vol.96, No.12, 2013, pp.2731–2734.

[4] Z. Ni, Y.Q. Shi, N. Ansari and W. Su, "Reversible data hiding", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 16, No. 3, March 2006,pp.354–362.

[5] C.C. Chang, C.C. Lin, C.S. Tseng and W.L. Tai, "Reversible hiding in DCT-based compressed images", Information Sciences, Vol. 177, No. 13, July 2007, pp. 2768–2786.

[6] X. Liao, K. Li and J. Yin, "Separable data hiding in encrypted image based on compressive sensing and discrete fourier transform", Multimedia Tools and Applications, Vol. 76, No. 20, 2016, pp. 20739–20753.

[7] B.Yang, M. Schmucker, W. Funk, C. Brush and S.Sun, "Integer DCT-based reversible watermarking for images using compounding technique", Proceedings of the SPIE, Security, Steganography and Watermarking of Multimedia Contents, vol. 5306, June 2004, pp. 405–415.

[8] C.C. Chang and C.Y. Lin, "Reversible steganographic method using SMVQ approach based on declustering", Information Sciences, Vol. 177, No. 8, 2007, pp.1796–1805.

[9] C.C. Chang, C.Y. Lin and Y.H. Fan, " Lossless data hiding for color images based on block truncation coding", Pattern Recognition, Vol. 41, No. 7, July 2008, pp.2347–2357.

[10] P.F. Shiu, W.L. Tai, J.K. Jan, C.C. Chang and C.C. Lin, "An Interpolative AMBTC-based high-payload RDH scheme for encrypted images", Signal Processing: Image Communication, Vol. 74, May 2019, pp. 64–77.

[11] C.K. Chan and L.M. Chen, "Hiding Data in Images by Simple LSB Substitution", Pattern Recognition, Vol. 37, No. 3, March 2004, pp.469–474.

[12] D.C. Wu and W.H. Tsai, "A steganographic method for images by pixel-value differencing", Pattern Recognition Letters, Vol. 24, No. 9, June 2003, pp.1613–1626.

[13] X. Liao, J. Yin, S. Guo, X. Li and A.K. Sangaiah, " Medical JPEG image steganography based on preserving inter-block dependencies", Computers & Electrical Engineering, Vol. 67, 2018, pp:320–329.

[14] A. Amirulhaqi, T.W. Purboyo and R.A.Nugrahaeni, "Security on GIF images using steganography with LSB method, spread spectrum and the vigenere cipher", International Journal of Applied Engineering Research, Vol. 12, No. 23, 2017, pp. 13604–13609.

[15] E. Delp and O. Mitchell, "Image compression using block truncation coding", IEEE Transactions on Communications, Vol. 27, No. 9, September 1979, pp. 1335–1341.

[16] J. Chen, W. Hong, T.S. Chen and C.W. Shiu, "Steganography for BTC compressed images using no distortion technique", The Imaging Science Journal, Vol. 58, 2010, pp. 177-185.

[17] C.H. Li, Z.M. Lu and Y.X. Su, "Reversible data hiding for BTC-compressed images based on bitplane flipping and histogram shifting of mean tables", Information Technology Journal, Vol. 10, No. 7, 2011, pp.1421–1426.

[18] W. Sun, Z.M. Lu and Y.C. Wen, "High performance reversible data hiding for block truncation coding compressed images", Signal Image Video Processing, Vol. 7, 2013, pp. 297–306.

[19] C.C. Lo, Y.C. Hu, W.L. Chen and C.M. Wu, "Reversible data hiding scheme for BTC-compressed images based on histogram shifting", International Journal of Security and its Applications, Vol. 8, No. 2, 2014, pp.301–314.

[20] I.C. Chang, Y.C. Hu, W.L. Chen and C.C. Lo, "High capacity reversible data hiding scheme based on residual histogram shifting for block truncation coding", Signal Processing, Vol. 108, 2015, pp.376–388.

[21] C.C. Lin, X.L. Liu, W.L. Tai and S.M. Yuan, "Novel reversible data hiding scheme based on AMBTC compression technique", Multimedia Tools and Applications, Vol.74, 2015,pp.3823–3842.

[22] C.C. Lin, C.C. Chang and Z.M. Wang, "Reversible data hiding scheme using adaptive block truncation coding based on an edge-based quantization approach", Symmetry, Vol. 11, 2019.

[23] M.D. Lema and O.R. Mitchell, "Absolute moment block truncation coding and its application to color image", IEEE Transactions on Communications, Vol. 32, 1984, pp.1148–1157.

[24] D. Ou and W.Sun, "High payload image steganography with minimum distortion based on absolute moment block truncation coding", Multimedia Tools and Applications, Vol. 74, 2015, pp.9117–9139.